scans to get more reliable measurements; using alternative strategies for picking the client in step 3, above, such alternative strategies including for example information theoretic methods based on, for example, various entropy or mutual information methods, which might enable picking the client leading to most information increase; using other grid cell shapes than rectangular such as for example hexagonal or circular cells, possibly allowing some overlap between adjacent cells.

[0065] The present embodiment is beneficial in, for example, a crowd-sourced video service. An example use case is related to a concert, where a crowd of users are going to. During the concert, the users shoot video of the event. After the concert, the video content is uploaded to the video service. The video service then creates an automatic cut of the video clips of the users. The video service also analyses sensory data captured by the mobile recording client to determine which are interesting points at each point in time during the event, and then makes switches between different source media in the final cut. Audio alignment is used to find a common timeline for all the source videos, and, for example, dedicated sensor data (accelerometer, compass) analysis algorithms are used to detect when several users are pointing to the same location on the stage, most likely indicating an interesting event. Furthermore, music content analysis (beats, downbeats) is used to find a temporal grid of potential cut points in the event soundtrack. When the user position is determined according to the previous embodiment, more interesting cuts can be obtained in the final output for the video service.

[0066] FIG. **3** shows an example of the crowd source video scenario. The event is being filmed by seven users (circles labelled A to G). A 3×3 grid is used to define the positions of the users. FIG. **4** shows how user feedback can be asked and given during the shooting of an event. At first (**10**), a client device B (**400**) shows a viewfinder image. Then (**20**), the user of client device B (**400**) is asked for a position to be indicated with a user interface input means in a user interface element (**410**) shown together the viewfinder image. On the following phase (**30**) the user of client device B (**400**) clicks on his/her position (**415**) in the user interface element. After giving the position, the user may continue filming (**40**) with his/her client device B (**400**).

[0067] The positions of the devices (**1** to **8**) with respect to a 3×3 grid is shown in FIG. **5**. FIG. **6** shows the estimated positions of the devices after fixing the position of device **4** and running one iteration of the positioning algorithm. It is noted, that after fixing the position of device **4**, the position estimates for devices **1**, **2**, **3**, **5**, **6** and **7** are wrong (shown with dash lines in FIG. **6**) and the estimate for device **8** is correct-like (shown with a square in FIG. **6**). It is realized that the position estimate needs not to be exactly correct, but can vary slightly around the correct position. FIGS. **7**, **8** and **9** show the position estimates after fixing additional device positions and running more iterations of the algorithms. After four iterations, the algorithm has found the correct position for all devices. This is shown in FIG. **9**.

[0068] In an embodiment, the algorithm runs until seven devices positions are known and then repeated eight times. For each repetition, a different device was used to ask the first position. FIG. **10** shows the average accuracy of the position estimates for the eight tests for each iteration of the algorithm. The gray line (lower line) shows the accuracy being calculated as the percentage of correctly positioned devices, whose positions are not yet fixed. The black line (i.e. the upper line) shows the accuracy over all devices (including the ones whose positions are not known). So for example, the average accuracy for the devices with unfixed positions after four iterations is 66%. This means that the position of four (fixed) devices are known and on average 66% of the position of the remaining four. Therefore, actually positions of 83% of all devices are known.

[0069] It is to be noticed that the algorithm, as described above, assumes that no devices are in the same grid cells as the "anchor devices" (see step 5 of the algorithm). This assumption can be removed by changing the definition of the "close" grid cells so that the cell of the "anchor device" is also considered as a "close" grid cell.

[0070] In the previous a method for relative positioning was explained so that the positioning algorithm was carried out by a server. It is to be noticed that the solution can also be performed without a server, so that one or more of the devises run the positioning algorithm. The signal strengths are thus gathered by a device/devices in question. In order to share the signal strength data between devices, a following information sharing method can be utilized.

[0071] This information sharing method uses a "FriendlyName" information that is transmitted by and between Bluetooth devices. Such a method can be used, for example, for relative positioning without client-server architecture. In a nutshell, the method works by having the devices change their Bluetooth "FriendlyNames" to convey information, such as signal strengths, to each other. The method is explained in more detail below.

[0072] Each Bluetooth device can transmit its name in a "FriendlyName" field, and this can be set by the user or the software. This name appears when other Bluetooth devices scan for surrounding Bluetooth devices.

[0073] In this information sharing method, the "FriendlyName" can be modified in a way that information can be shared to other users. Some different embodiments are presented below:

[0074] In the first embodiment the client device may scan all the nearby device names, append them along with the received signal strength indicator value to its own "FriendlyName". For example: FriendlyName: Device**1**: Device**2**; −95 dBm: Device**3**; −90 dBm. Such a name will then appear when other devices scan Device **1**. The information in the "FriendlyName" can be used to solve the relative positions of each device, as all the link strengths become visible for all the devices. Device**1** alone cannot determine the link strength between Device**2** and Device**3**, but it can see it from the modified FriendlyName of Device**2** or Device**3**. This embodiment can be utilized by the method for relative positioning being disclosed above so that no server is needed for making the positioning calculations.

[0075] In the second embodiment the client may scan all the nearby device names, append them along with some information field, such as classified motion mode of each device, to its own "FriendlyName". For example: FriendlyName: Device**1**; standing: Device**2**; walking: Device**3**; table. This way the devices that cannot hear Device**3** (for example) can still see the information via the "FriendlyName" of Device**1**. In addition to the motion mode of the devices, also user indicated position information such as the one being used in the method for relative positioning disclosed above can be sent between devices.